Short vs. Long Flows: A Battle That Both Can Win

Morteza Kheirkhah, Ian Wakeman and George Parisis
School of Engineering and Informatics
University of Sussex, UK
{m.kheirkhah, ianw, g.parisis}@sussex.ac.uk

ABSTRACT

In this paper, we introduce MMPTCP, a hybrid transport protocol which aims at unifying the way data is transported in data centres. MMPTCP runs in two phases; initially, it randomly scatters packets in the network under a single congestion window exploiting all available paths. This is beneficial to latency-sensitive flows. During the second phase, MMPTCP runs in Multi-Path TCP mode, which has been shown to be very efficient for long flows. Initial evaluation shows that our approach significantly improves short flow completion times while providing high throughput for long flows and high overall network utilisation.

1. INTRODUCTION

Modern data centres [1, 3] provide very high aggregate bandwidth and multiple paths among servers. They support a large number of services which produce very diverse intra-data centre traffic matrices. Long flows are bandwidth hungry, while short ones commonly come with strict deadlines regarding their completion time. It has been shown that TCP is ill-suited for both types of traffic in modern data centres, where ECMP [4] is used to exploit the availability of multiple equal-cost paths. Under high load, long flows collide with high probability and, as a result, network utilisation significantly drops and only 10% of the flows achieve their expected throughput [6]. TCP is also inefficient for short flows, especially when competing with long flows. Queue build-ups, buffer pressure and TCP Incast combined with the shared-memory nature of data centre switches results in short TCP flows missing their deadlines mainly due to retransmission timeouts (RTOs) [2].

To cope with these challenges, several transport protocols have been recently proposed. DCTCP [2], D2TCP [7] and D3 [8] aim at minimising flow completion times for latency-sensitive flows. However, they require modifications in the network and/or deadline-awareness at the application layer. Such modifications are problematic because such information may not be known a priori (i.e. at connection time). More importantly, all of these protocols are single-path and thus cannot exploit the multipath potential of data centre networks. Multipath transport protocols, such as Multi-Path TCP [6], transport data using multiple sub-flows and rely on ECMP to achieve higher aggregated throughput over multiple paths. As shown in [6], MPTCP improves the overall network utilisation. However, MPTCP hurts short flows as the number of sub-flows increases. The congestion window of a sub-flow may be very small over the sub-flow lifetime and, as a result, even a single lost packet can force an entire connection to wait for an RTO to be triggered because the lost packet cannot be recovered through fast retransmission. This is clearly illustrated in Figure 1(a)[1], where the mean flow completion time of short flows increases as more sub-flows are used (better shown in the embedded Figure in Figure 1(a)). Note that the number of connections that experience one or more RTOs significantly increases as well, hence the increase in the standard deviation. Even a single RTO may result in flow deadline violation.

Supporting and running multiple transport protocols in a data centre can be problematic. Fairness among different protocols is difficult to achieve; most protocols for latency-sensitive flows are not compatible with TCP or MPTCP [2, 7]. Running multiple transport protocols is also a burden for application developers who would have to decide upon the most suitable transport protocol. Both application requirements and data centre topologies evolve over time and so a transport protocol that performs well over disparate topologies and traffic matrices is a necessity.

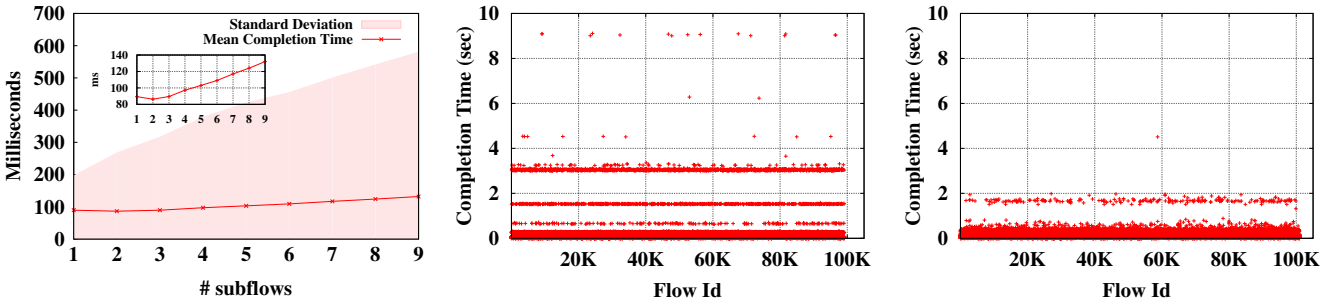
In this paper, we introduce MMPTCP, a hybrid transport protocol that aims at unifying data transport within data centres. MMPTCP objectives are: (1) high throughput for large flows, (2) low latency for short flows, and (3) tolerance to sudden and high bursts of traffic, all without application-layer information. Co-existence in harmony with legacy TCP and MPTCP flows is also an objective.[2] Data transport takes place in two phases. Initially, packets are randomly scattered in the network under a single TCP congestion window exploiting all available paths. Most, if not all, short flows are expected to complete before switching to the second phase, during which, MMPTCP runs as standard MPTCP, efficiently handling long flows.

2. MMPTCP DESIGN

Packet Scatter (PS) Phase. The PS protocol applicability to data centres has been briefly explored in [6], where it has been shown that it can eliminate network congestion at the network core. In our approach packet scattering is initiated at the end hosts through source port randomisation rather than switches. Network switches then forward packets to all available paths employing hash-based ECMP. The main challenge here is the graceful handling of out-of-order packets. We are currently exploring the following approaches: (1) Dynamically assigning the duplicate ACK threshold using topology-specific information. For example,

[1]All simulations are based on our custom implementation of MPTCP and MMPTCP in ns-3 [5]

[2]Our initial results shows that our objectives are achievable



(a) MPTCP (b) MPTCP (8 subflows) (c) MMPTCP (PS and 8 subflows)

Figure 1: Short flow completion times in a simulated 4:1 over-subscribed FatTree topology consisting of 512 servers and supporting a 4:1 over-subscription ratio. One third of the servers run long (background) flows. The rest run short flows (70KBs each) which are scheduled according to a Poisson process). All flows are scheduled based on a permutation traffic matrix.

FatTree's IP addressing scheme can be exploited to calculate the number of available paths between the sender and receiver. Other data centre topologies, such as VL2, incorporate centralised components which can provide similar information. (2) Alternatively, approaches such as RR-TCP [9] could be used to minimise spurious retransmissions when out-of-order packets are misidentified as lost.

Phase Switching. Switching to MPTCP at the right time is crucial to ensuring that short flows complete very fast before switching, while long flows are not hurt by running with a single congestion window for long. We are currently investigating two switching strategies. *(1) Data Volume:* Switching occurs when a certain amount of data has been transmitted. Early evaluation suggests that this approach does not exert any negative effects on the throughput of long flows since the opening of multiple sub-flows after switching can wrap up access link capacity in a few RTTs. *(2) Congestion Event:* Switching occurs when congestion is first inferred (i.e. when fast retransmission or an RTO is triggered).

MPTCP Phase. When the switching threshold is reached, MMPTCP initiates a number of sub-flows and data transport is governed by MPTCP's congestion control. No more packets are put in the initial PS flow which is deactivated when its window gets emptied.

3. DISCUSSION AND FUTURE WORK

Figures 1(b) and 1(c) depict the flow completion times for all short flows in a simulated FatTree topology (see Figure 1 caption for details of the simulation setup). It is clear that with MPTCP (1(b)) a lot more short flows experience one or more RTOs leading to very high completion times, compared to MMPTCP (1(c)). During the packet scatter phase, MMPTCP utilises all available paths to distribute packets to receivers, while using a single congestion window, gracefully handling sudden bursts. Note that the majority of short flows completed within 100ms. The average flow completion time and the standard deviation for MMPTCP and MPTCP are 116 milliseconds (standard deviation is 101) and 126 milliseconds (standard deviation is 425), respectively. Although not shown in the figure, we can report that with MMPTCP the average loss rate at the core and aggregation layers are slightly lower compared to MPTCP and both protocols achieve the same average throughput for long flows and overall network utilisation.

Roadmap. Using our custom ns-3 models for MPTCP and MMPTCP, we are currently simulating several data centre topologies, comparing MMPTCP to other transport protocols in a wide range of network scenarios (e.g. effect of hotspots, network loads, traffic matrices and phase switching strategies). We also plan to design multi-homed network topologies as these are well-suited to MMPTCP. The more parallel paths at the access layer, the higher the burst tolerance; hence, potentially, preventing transient congestion. Our early results are promising and we expect that we will soon be able to report on this wider range of experiments.

We expect that MMPTCP will be readily deployable in existing data centres as it can coexist with other transport protocols. In-depth investigation of how MMPTCP shares network resources with TCP and MPTCP is part of our current work. Early results suggest that it could co-exist in harmony with them. MMPTCP requires ECMP, which is deployed in all data centres, and doesn't rely on any changes in the network or any application-layer information (e.g. flow size and deadline).

4. REFERENCES

[1] M. Al-Fares et al. A Scalable, Commodity Data Center Network Architecture. In *Proc. of SIGCOMM 2008.*

[2] M. Alizadeh et al. Data Center TCP (DCTCP). In *Proc. of SIGCOMM 2010.*

[3] A. Greenberg et al. VL2: A Scalable and Flexible Data Center Network. In *Proc. of SIGCOMM 2011.*

[4] C. Hopps. Analysis of an equal-cost multi-path algorithm. RFC 3782, 2004.

[5] M. Kheirkhah et al. Multipath TCP model in ns-3. In WNS3 2014, `https://github.com/mkheirkhah/mptcp`.

[6] C. Raiciu et al. Improving Datacenter Performance and Robustness with Multipath TCP. In *Proc. of SIGCOMM 2011.*

[7] B. Vamanan et al. Deadline-aware Datacenter TCP (D2TCP). In *Proc. of SIGCOMM 2010.*

[8] C. Wilson et al. Better Never than Late: Meeting Deadlines in Datacenter Networks. In *Proc. of SIGCOMM 2011.*

[9] M. Zhang et al. RR-TCP: A Reordering-Robust TCP with DSACK. In *Proceedings of ICNP 2003.*